We claim:

1. A machine-readable medium that provides instructions, which when executed by a set of processors of one or more processors, cause said set of processors to perform operations comprising:

storing an operation; and

executing the operation idempotently with a network resource process.

2. The machine-readable medium of claim 1 wherein storing the operations comprises:

storing the operation in a log as a record;

receiving a commit command; and

moving the record into an atomic database.

3. The machine-readable medium method of claim 1 wherein storing the operation comprises:

receiving the operation;

performing lock contention handling for the operation;

storing the operation if a lock contention is not detected; and

generating a lock contention notification if the lock contention is detected for the operation.

4. The machine-readable medium of claim 1 wherein the operation is one of a sequence of operations comprising an atomic transaction.

5. The machine-readable medium of claim 1 further comprising:

receiving the operation from a first user; and

receiving a second operation from a second user.

6. The machine-readable medium of claim 1 wherein the operation is received from a first user concurrently with a second operation received from a second user.

7. A machine-readable medium that provides instructions, which when executed by a set of processors of one or more processors, cause said set of processors to perform operations comprising:

storing a sequence of operations; and

performing the sequence of operations as an atomic transaction.

8. The machine-readable medium of claim 7 wherein each of the sequence of operations is performed idempotently.

9. The machine-readable medium of claim 7 wherein storing the sequence of operations comprises:

performing lock contention handling for each of the sequence of operations;

storing the sequence of operations if a lock contention is not detected; and

generating a lock contention notification if the lock contention is detected.

10. The machine-readable medium of claim 7 wherein a sequence of operations is received from a first user and a second sequence of operations is received from a second user.

11. The machine-readable medium of claim 7 wherein the sequence of operations is received from a first user in concurrence with receiving a second sequence of operations from a second user.

12. A machine-readable medium that provides instructions, which when executed by a set of processors of one or more processors, cause said set of processors to perform operations comprising:

storing an operation in an atomic database; and

performing the operation with a network resource process in response to an commit command.

13. The machine-readable medium of claim 12 wherein the operation is performed idempotently.

14. The machine-readable medium of claim 12 wherein the operation is one of a sequence of operations comprising a transaction.

15. The machine-readable medium of claim 12 wherein storing the operation in the atomic database comprises:

performing lock contention handling for the operation;

storing the operation in the atomic database if a lock contention is not detected; and

generating a notification of lock contention if the lock contention is detected.

16. The machine-readable medium of claim 12 wherein the operation is received from a first user and a second operation is received from a second user.

17. The machine-readable medium of claim 12 wherein the operation is received from a first user in concurrence with a second operation received from a second user.

18. A machine-readable medium that provides instructions, which when executed by a set of processors of one or more processors, cause said set of processors to perform operations comprising:

receiving an operation;

determining if a lock contention exists for a record corresponding to the operation; and

generating a notification of the lock contention if a lock contention does exist for the record.

19. The machine-readable medium of claim 18 wherein the operation is one of a sequence of operations comprising an atomic transaction.

20. The machine-readable medium of claim 18 wherein the operation is performed idempotently with a network resource process.

21. The machine-readable medium of claim 18 wherein the operation is received from a first user and a second operation is received from a second user.

22. The machine-readable medium of claim 18 wherein the operation is received from a first user in concurrence with a second operation received from a second user.

23. The machine-readable medium of claim 18 further comprising storing the operation if the lock contention does not exist.

24. A machine-readable medium that provides instructions, which when executed by a set of processors of one or more processors, cause said set of processors to perform operations comprising:

storing a first and second operation as an atomic transaction;

performing the first and second operation idempotently with a set of network resource processes.

25. The machine-readable medium of claim 24 wherein storing the first and second operation as an atomic transaction comprises:

performing lock contention handling for the first and second operation;

storing the first and second operation of the atomic transaction in a log if a lock contention is not detected; and

generating a lock contention notification if the lock contention is detected.

26. The machine-readable medium of claim 24 wherein storing the first and second operation as the atomic transaction comprises:

Storing the first and second operation of the atomic transaction in a log;

receiving a commit command for the atomic transaction;

indicating the atomic transaction as committed; and

storing the atomic transaction in an atomic database.

27. The machine-readable medium of claim 24 wherein the first and second operation are received from a first user and a third operation of a second transaction is received from a second user.

28. The machine-readable medium of claim 24 wherein the second operation is received from a first user concurrently with a third operation received from a second user.

29. A network element comprising:

a processor to execute a set of atomic transactions; and

a storage unit coupled to the processor, the storage unit to store the set of atomic transactions.

30. The network element of claim 29 wherein each of the set of atomic transactions is comprised of a set of operations, the processors to execute each of the set of operations idempotently.

31. The network element of claim 29 wherein the processor forms lock contention handling for the set of atomic transactions; and the processor generates the lock contention notification if the lock contention is detected.

32. The network element of claim 29 further comprising a set of interfaces coupled to the processor, each of the set if interfaces to receive at least one of the set of atomic transactions, each of the set of interfaces corresponding to a different user.

33. A network element comprising:

an interface to receive a plurality of operations from a user;

a configuration manager coupled to the interface, the configuration manager to process the plurality of operations; and

an atomic database coupled to the configuration manager, the atomic database to store the plurality of operations as a transaction.

34. The network element of claim 33 wherein the configuration manager processes the plurality of operations as an atomic transaction.

35. The network element of claim 33 further comprising:

the atomic database to detect lock contention;

the configuration manager to generate a notification of the lock contention detected by the atomic database; and

the interface to display a message corresponding to the notification generated by the configuration manager.

36. The network element of claim 33 further comprising a second interface coupled to the configuration manager, the second interface to receive a second plurality of operations from a second user.

37. A computer implemented method comprising:

storing an operation; and

executing the operation idempotently with a network resource process.

38. A computer implemented method of claim 37 wherein storing the operations comprises:

storing the operation in a log as a record;

receiving a commit command; and

moving the record into an atomic database.

39. The computer implemented method of claim 37 wherein storing the operation comprises:

receiving the operation;

performing lock contention handling for the operation; storing the operation if a lock contention is not detected; and

generating a lock contention notification if the lock contention is detected for the operation.

40. The computer implemented method of claim 37 wherein the operation is one of a sequence of operations comprising an atomic transaction.

41. The computer implemented method of claim 37 further comprising:

receiving the operation from a first user; and

receiving a second operation from a second user.

42. The computer implemented method of claim 37 wherein the operation is received from a first user concurrently with a second operation received from a second user.

43. A computer implemented method comprising:

storing a sequence of operations; and

performing the sequence of operations as an atomic transaction.

44. The computer implemented method of claim 43 wherein each of the sequence of operations is performed idempotently.

45. The computer implemented method of claim 43 wherein storing the sequence of operations comprises:

performing lock contention handling for each of the sequence of operations;

storing the sequence of operations if a lock contention is not detected; and

generating a lock contention notification if the lock contention is detected.

46. The computer implemented method of claim 43 wherein a sequence of operations is received from a first user and a second sequence of operations is received from a second user.

47. The computer implemented method of claim 43 wherein the sequence of operations is received from a first user in concurrence with receiving a second sequence of operations from a second user.

48. A computer implemented method comprising:

receiving an operation;

determining if a lock contention exists for a record corresponding to the operation; and

generating a notification of the lock contention if a lock contention does exist for the record.

49. The computer implemented method of claim 48 wherein the operation is one of a sequence of operations comprising an atomic transaction.

50. The computer implemented method of claim 48 wherein the operation is performed idempotently with a network resource process.

51. The computer implemented method of claim 48 wherein the operation is received from a first user and a second operation is received from a second user.

52. The computer implemented method of claim 48 wherein the operation is received from a first user in concurrence with a second operation received from a second user.

53. The computer implemented method of claim 48 further comprising storing the operation if the lock contention does not exist.